# Applied Data Science Capstone : Coursera

## Tanish Ghosh

# tanishghosh@gmail.com

## Data

The data required for this project have been acquired from multiple sources. Any error or discrepancy of the data found is credited to the source only.

## Neighborhood Data

The neighborhood names of Mumbai are extracted from a .csv file containing all the pincodes of India(Source: https://data.gov.in/resources) along with other details using Pandas library for Python. The neighborhood names and their corresponding pincode is stored in a Pandas DataFrame.

**Code**

import pandas as pd

df = pd.read\_csv('PO.csv')

neighborhood=[]

for i in df.index:

district = df['regionname'][i]

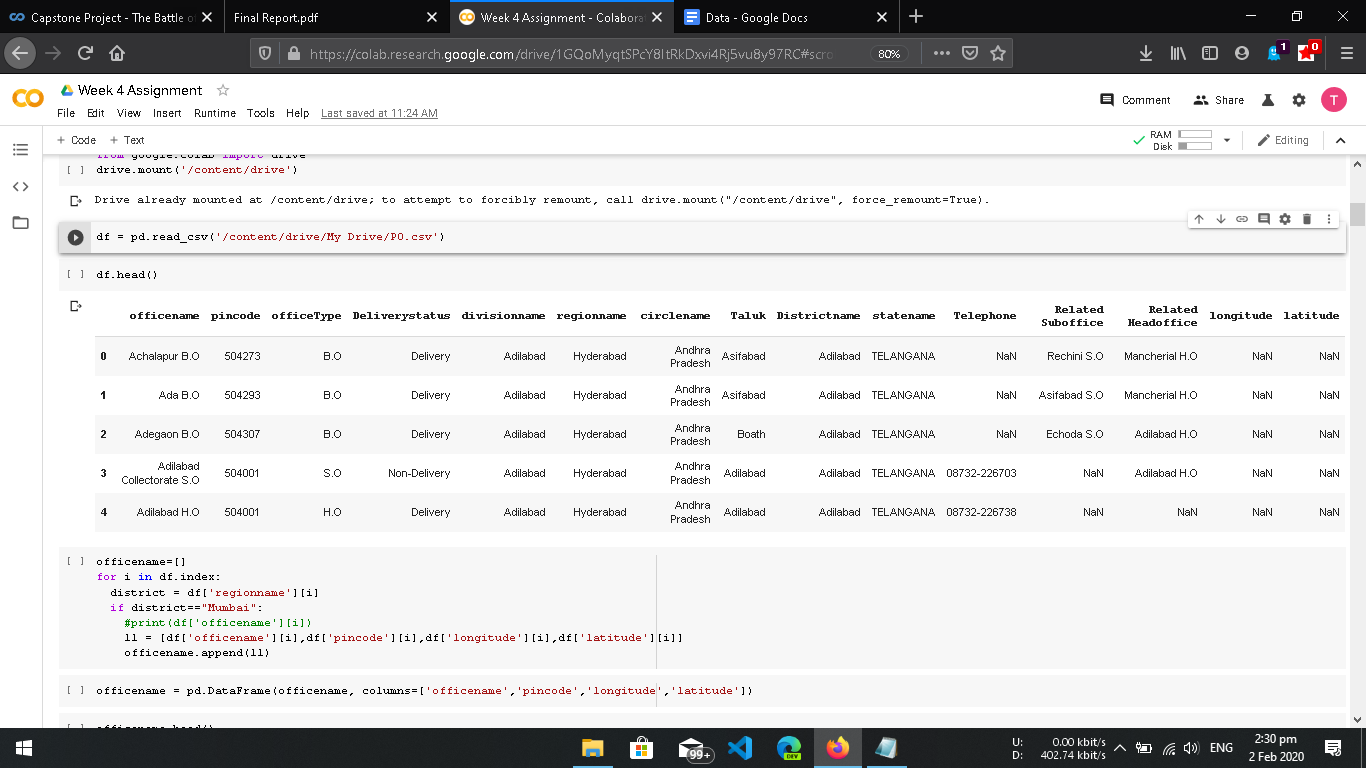
if district=="Mumbai":

l1 = [df['officename'][i],df['pincode'][i]]

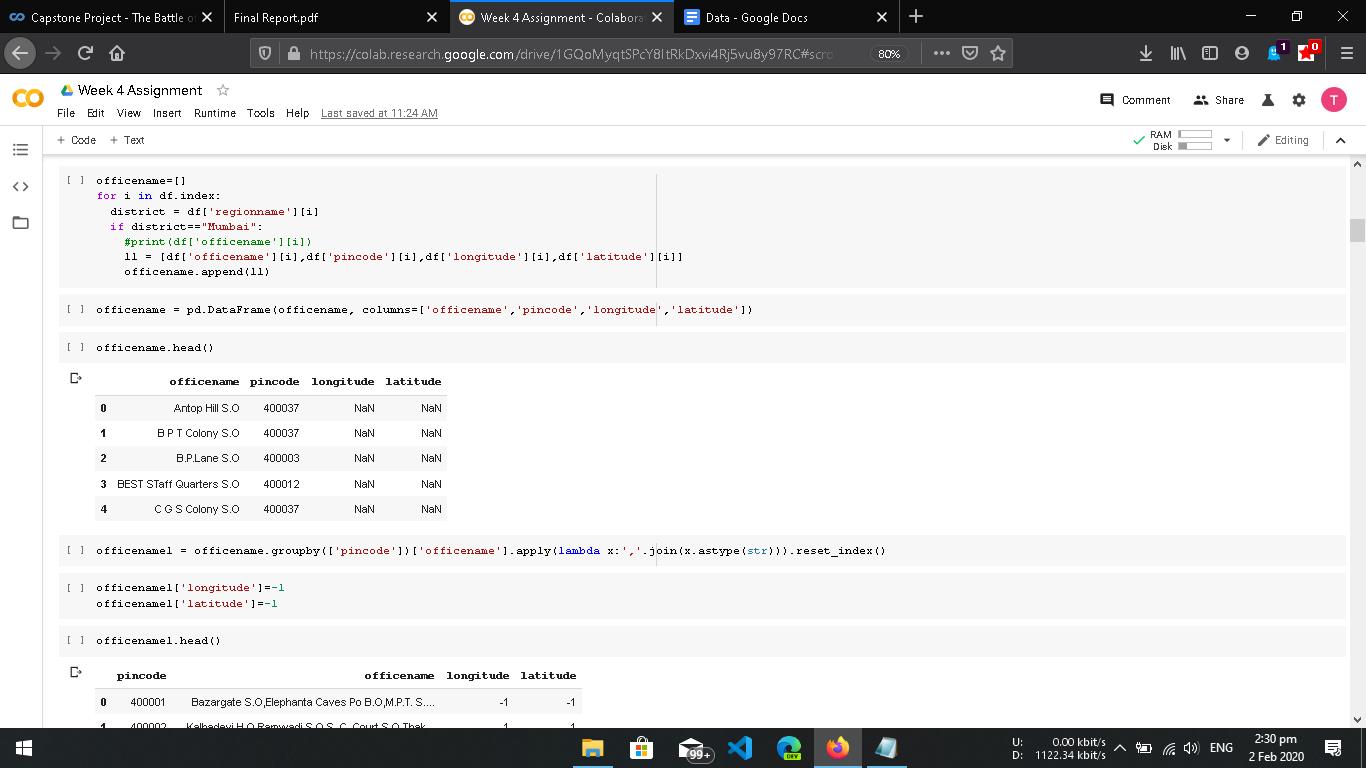
neighborhood.append(l1)

neighborhood = pd.DataFrame(neighborhood, columns=['officename','pincode'])

Initial dataset



Neighborhoods of Mumbai

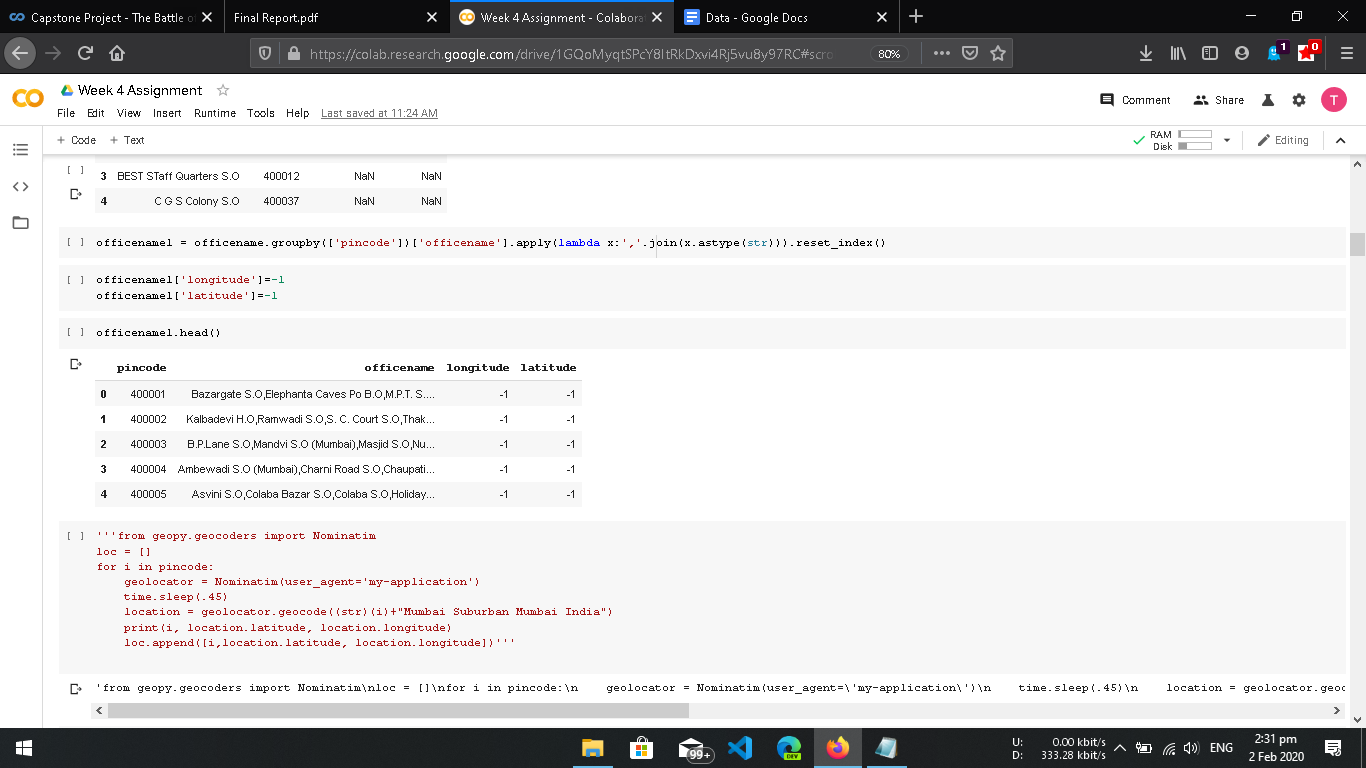


We then group neighborhoods having same pincode in order to remove redundancies.

**Code**

neighborhood1=neighborhood.groupby(['pincode'])['officename'].apply(lambda x:','.join(x.astype(str))).reset\_index()

Neighborhoods of Mumbai (after grouping by pincode)



## Coordinates of Neighborhoods

The latitude and longitude of the neighbourhoods are retrieved using geopy library. Geopy locate the coordinates of addresses, using third-party geocoders and other data sources. The geometric location values are then stored into the initial dataframe.

**Code**

import geopy

from geopy.geocoders import Nominatim

loc = []

pincode = neighborhood1[‘pincode’]

for i in pincode:

geolocator = Nominatim(user\_agent='my-application')

time.sleep(.45)

location = geolocator.geocode((str)(i)+"Mumbai India")

print(i, location.latitude, location.longitude)

loc.append([i,location.latitude, location.longitude])

temp = pd.DataFrame(loc)

temp.columns = [‘pincode’,’longitude’,’latitude’]

neighborhood1['longitude']=temp['longitude']

neighborhood1['latitude']=temp['latitude']

## Venue Data

From the location data, the venue data is found out bypassing in the required parameters to the FourSquare API, and creating another DataFrame to contain all the venue details along with the respective neighborhoods.

**Code**

def getNearbyVenues(names, latitudes, longitudes, radius=500):

LIMIT = 15

venues\_list=[]

for name, lat, lng in zip(names, latitudes, longitudes):

print(name)

url = 'https://api.foursquare.com/v2/venues/explore?&client\_id={}&client\_secret={}&v={}&ll={},{}&radius={}&limit={}'.format(

CLIENT\_ID,

CLIENT\_SECRET,

VERSION,

lat,

lng,

radius,

LIMIT)

# make the GET request

results = requests.get(url).json()["response"]['groups'][0]['items']

# return only relevant information for each nearby venue

venues\_list.append([(

name,

lat,

lng,

v['venue']['name'],

v['venue']['location']['lat'],

v['venue']['location']['lng'],

v['venue']['categories'][0]['name']) for v in results])

nearby\_venues = pd.DataFrame([item for venue\_list in venues\_list for item in venue\_list])

nearby\_venues.columns = ['Neighborhood','Neighborhood Latitude', 'Neighborhood Longitude', 'Venue', 'Venue Latitude','Venue Longitude', 'Venue Category']

return(nearby\_venues)

mumbai\_venues = getNearbyVenues(names=neighborhood1['officename'],

latitudes=neighborhood1['latitude'],

longitudes=neighborhood1['longitude']

)

# 